IDENTIFYING A FAILED DEVICE IN A NETWORK

Background of the Invention

The present invention relates to identifying a failed device on a network.

Data packets on a network are transmitted through devices, such as routers and switches, on the way to their destinations. In many networks, these devices are connected in such a way that failure of one device can make an entire branch of the network inaccessible. For example, if a device is the only route to several other devices, failure of that device will prevent data packets from reaching those other devices and their downstream destinations.

To restore full network service, a failed device must be identified and repaired. The failed device can be identified by determining the topology of the entire network and then locating the failed device using the topology. This approach, however, can be computationally intensive, particularly where the network topology changes over time.

Summary of the Invention

One aspect of the invention identifies a failed network device based on information obtained from the device's neighbors. In particular, each device on the network stores information, such as Management Information Base ("MIB") II tables, that specifies which other devices are connected to the device. This information is compiled and used to generate a neighbor table for the network.

In the event of a device failure, the neighbor table is used to identify which device failed. More specifically, a packet is sent to a target device in order to determine if the target device is active. If the target device is not

active, the neighbor table is consulted in order to locate devices that neighbor the target device. It is then determined if any of the target device's neighbors are active. If so, the target device is identified as being a failed device. If not, the process is repeated for the neighbors until an active device is located.

The various embodiments of the invention possess one or more of the following advantages. First, the user can identify failed network devices without having to determine the entire network topology. Stated differently, the embodiments enable the user to identify failed devices on the network by retrieving significantly less information about the network than has been previously thought necessary. Another benefit of the various embodiments is they can be used from any vantage point on the network without requiring hardware or software reconfiguration.

In general, in one aspect, the invention identifies a failed device in a network that includes plural devices. This aspect includes attempting to communicate with a target device and determining if the target device has an active neighbor if the attempt to communicate with the target device fails. The target device is identified as a failed device if the target device has an active neighbor.

In other aspects, the invention includes one or more of the following features/functions. The attempting comprises sending a packet to the target device and waiting for a response from the target device. The determining comprises attempting to communicate with a neighbor of the target device. The neighbor is determined to be active if the attempt to communicate is successful.

The neighbor of the target device is located by generating a neighbor table for the network and consulting the neighbor table. Generating comprises polling the target

device, receiving a response from the target device, and constructing the neighbor table based on the response. The polling is performed periodically. The neighbor table is updated based on the periodic polling. The response comprises a network address of the neighbor. The neighbor table indexes the target device to the network address of the neighbor. The target device stores a MIB II (or other type of) table containing the network address of the neighbor. The target device prepares the response based on the MIB II table. The target device and the neighbor can comprises a router, a switch, a server, a personal computer, or any other type of networked device.

In general, in another aspect, the invention identifies a failed device in a network that includes plural devices by generating a neighbor table for the devices based on information provided from the devices and sending a packet to a target device to determine if the target device is active. If the target device is not active, a neighbor of the target device is located using the neighbor table and a packet is sent to the neighbor to determine if the neighbor is active. The target device is identified as a failed device if the neighbor is active.

In general, in another aspect, the invention features a network system that includes first, second and third devices, where the third device is located in a path between the first and second devices. The first device includes a processor and memory which stores executable code. The processor executes code to send a packet to the second device to determine if the second device is active. If the second device is not active, the processor sends a packet to the third device to determine if the third device is active. The processor identifies the second device as a failed device if the third device is active.

Advantages of the invention in addition to those set forth above will become apparent in view of the following description, including the claims and the drawings.

Brief Description of the Drawings

Fig. 1 shows a network topology.

Fig. 2 shows a process for identifying a failed device in the network.

Fig. 3 shows a process for generating a neighbor table for the network.

Fig. 4 shows an alternative network topology on which the invention may be used.

Description of the Preferred Embodiment

Fig. 1 shows a network 1 on which an embodiment is implemented. Network 1 may be any kind of network, such as a local area network ("LAN"), a wide area network ("WAN"), or the Internet. Components of network 1 include hosts 2 and 8, routers 3 to 9, and switches 10 to 12.

Routers 3 to 9 are computing devices for routing data packets through network 1 based on the packets' Internet Protocol ("IP") addresses. Each router includes a processor 14 and a memory 15 which stores routing tables 16 and routing code 17 (see view 19 of router 4). Code 17 is executed by processor 14 to route data packets through network 1. The information stored in routing tables 16 is used to perform the routing.

Routing tables 16 include MIB II tables 20 (see "Management Information Base For Network Management Of TCP/IP-Based Internets: MIB II", RFC 1213 (March 1991)). In particular, MIB II tables 20 are "atNetAddress" tables which contain the IP addresses of devices that neighbor the router. For example, in router 4 MIB II tables 20 contain

the IP addresses of routers 3, 5 and 6, and the IP addresses of each interface (or "socket") to routers 3, 5 and 6 (each router has a device IP address and each socket on that router can have a separate IP address).

MIB II tables 20 also include network address conversion tables 21, examples of which are the MIB II "Address Translation" tables and "ipNetToMediaTable". The conversion tables in each router store the MAC and IP addresses of devices that neighbor the router and are used to convert MAC addresses to IP addresses, and vice versa, during routing.

Switches 10 to 12 are electronic devices for routing data packets by device address. The IP/MAC address conversion tables in the routers perform the network address conversions required to route data packets between switches and routers. Each switch includes a memory 22 which stores MIB II tables 24 (see view 25 of switch 10). In some switches, MIB tables 24 contain "atNetAddress" tables which identify the switch's neighbors by IP address. In other switches, however, MIB tables 24 identify the switch's neighbors by MAC address, and not by IP address. Specifically, these switches use the MIB II "ifPhysAddress" table to identify their neighbors.

Host 2 is a personal computer ("PC") or similar device that is capable of communicating on network 1. Included in host 2 are the following: network connection 26 for interfacing to network 1, display screen 27 for displaying information to a user, keyboard 29 for inputting text and commands, mouse 30 for positioning a cursor on display screen 27 and inputting user commands, and drive 31 for accessing data stored on a computer-readable storage medium such as a computer diskette, a CD-ROM, or a DVD.

4

View 32 shows the architecture of host 2. Included in host 2 are keyboard interface 34, mouse interface 35, display interface 36, drive interface 37, RAM 39, processor 40, and memory 41. Memory 41 stores code 42 and a network neighbor table 44. Network neighbor table 44 identifies each device on network 1 (such as a router, switch or PC) and the neighbors of that device. Specifically, network neighbor table 44 indexes the IP address of each device to the IP addresses of its neighboring devices.

Code 42 (which, alternatively, may be stored in a storage medium in drive 31) is executed by processor 40 out of RAM 39. Code 42 includes failure identifier 45, network communication software 46, and operating system 47. Operating system 47 is a windowing operating system, such as Microsoft® WindowsNT®; however, other types of operating systems may be used. Network communication software 46 includes IP protocol stack layers and other necessary code for transmitting data packets to, and receiving data packets from, network 1. Failure identifier 45 detects a failure in network 1 and identifies the failed device in accordance with the process of Fig. 2.

Prior to that process, however, host 2 generates/updates a neighbor table for network 1 according to the process of Fig. 3. To begin, host 2 polls 301 devices 49 on network 1 using network communication software 46. Polling is performed periodically (e.g., every hour, five minutes, or less) by sending a data packet to the IP address of each device and then waiting for a response.

In 302, host 2 receives a response from each active device on the network. This response is prepared by each device based on the information contained in the device's MIB II tables. The response contains an IP address of the polled device, together with the IP addresses of its

neighbors. The IP addresses in the response could be device IP addresses. For example, when polled in 303, router 4 may return its device IP address, the device IP address of router 3, and device addresses of routers 5 and 6.

Switches, such as switch 11, may return MAC addresses of their neighbors, instead of IP addresses, for reasons noted above. Conversion tables in host 2 or in an upstream router (relative to switch 11) are used to convert switch MAC addresses into IP addresses. Verification of device neighbors may also be performed at host 2 using MIB II "atNetatPhysAddress" and "ifPhysAddress" tables.

Host 2 then determines 303 whether there is an existing network neighbor table for network 1. If so, host 2 updates 304 the existing table based on responses received in 302. Specifically, host 2 determines if the information in the responses contradicts information in the existing network neighbor table and, if so, substitutes that information with the information from the responses. If there is no existing network neighbor table, host 2 constructs a new network neighbor table based on the information received in 302, and stores it in memory 41.

Once a current network neighbor table is in place, the process of Fig. 2 is executed to identify failed devices on network 1. The process of Fig. 2 may be performed periodically, such as every five minutes, without regard to whether an outage in the network has been detected. Alternatively, the process may be performed upon detection of an outage. An outage may be detected in any number of ways. For example, host 2 may send a data packet to a destination and receive a message indicating that the packet is "undeliverable" which indicates a network outage.

In 201, host 2 selects a target device on network 1. Network neighbor table 44 defines the active devices on

network 1. Therefore, the target device is selected from the devices listed in network neighbor table 44 including, but not limited to, routers, switches, and other hosts. The devices may be selected in any order.

Next, host 2 attempts 202 to communicate with the target device. The attempt is performed by sending a packet to the target device and waiting for a response (this is called "pinging" the target device). If the target device responds, the attempt to communicate has been successful 203, meaning that the target device is active and, therefore, not the root cause of an outage (where "root cause" refers to the first failed device on a path from host 2 to a desired network destination). In this case, host 2 outputs an indication 204 that the target device is not the root cause of the outage, selects 201 a new target device, and repeats 202 and 203. If the target device does not respond, host 2 locates 205 the neighbor(s) of the target device by consulting network neighbor table 44.

Host 2 then determines 206 if any of the neighbors are active. This is done by sending a data packet to each neighbor and then waiting for a response. If there are no responses from the neighbors, there are no active devices interfaced to the target device, which means that the target device is not the root cause of the outage. In this case, host 2 outputs an indication that the target device is not the root cause of the outage. Host 2 then selects 201 a new target device and repeats 202 to 206. If, however, there is a response from one of the target device's neighbors, there is an active communication path between host 2 and the target device, which means that the target device is the root cause of the outage. Accordingly, host 2 identifies 207 the target router as the root cause of the outage. Host 2 then outputs 208 an indication (such as a display, an

alarm, a page, or an electronic mail) that the target router is the root cause. If all devices have been considered 209 (see below), the process ends; otherwise it returns to 201.

By way of example, assume that router 4 (see Fig. 1) is the root cause of an outage that affects routers 5 to 9 and switches 10 to 12. If router 7 is selected 201 as the target device, host 2 attempts 202 to communicate with router 7. Communication will be unsuccessful 203, since router 7 is part of the outage. Therefore, host 2 locates 205 neighbors of router 7 and determines 206 if any of them are active. Router 7's only neighbors are router 6 and switch 10. Both of these were also affected by the outage. Therefore, there is no active communication path between host 2 and router 7, which means that router 6 is not the root cause of the outage.

If router 6 is next selected 201 as the target device, host 2 attempts 202 to communicate with router 6. Communication will be unsuccessful 203 since router 6 is part of the outage. Therefore, host 2 locates 205 router 6's neighbors and determines 206 if any of them are active. Router 6's only neighbors are routers 4, 5, 7 and 8. These were all affected by the outage. Therefore, router 6 is not the root cause of the outage.

If router 4 is next selected 201 as the target device, host 2 attempts 202 to communicate with router 4. Communication will be unsuccessful 203 since router 4 is part of the outage. Therefore, host 2 locates router 4's neighbors 205 and determines 206 if any of them are active. Router 4's neighbors are routers 3, 5 and 6. Routers 5 and 6 are not active, since they are downstream from the outage relative to host 2. However, router 3 is active, which means that there is an active communication path between host 2 and router 3. Accordingly, router 4 is the first

inactive router downstream from host 2 and thus is the root cause of the outage.

Fig. 4 shows another network on which the invention may be used. Network 50 includes host 51, routers 52 to 57, and switch 58. These device are the same, in both structure and function, as corresponding devices of Fig. 1.

In network 50, an outage has left routers 54 to 57 and switch 58 inactive. To find the root cause of the outage, the process of Fig. 2 is executed. This process eventually results in host 51 selecting 201 router 56 as the target router. From there, host 51 attempts to communicate 202 with router 56. These attempts will be unsuccessful 203 since router 56 was affected by the outage; therefore, host 51 will locate 205 the neighbors of router 56. Router 56's neighbors are switch 58 and router 53. By trying to communicate with the neighbors, host 51 determines 206 that router 53 is active and that switch 54 is not active. Since router 56 has an active neighbor, router 56 is identified as the root cause of the failure in 207 and an indication to that effect is output in 208.

However, the process does not end there. In 210, host 2 determines whether all network devices have been considered. If not, as we assume here, the process returns to 201. From there, the process eventually results in host 51 selecting 201 router 54 as the target router. Host 51 therefore attempts to communicate 202 with router 54. These attempts will be unsuccessful 203 since router 54 was affected by the outage; therefore, host 51 will locate 205 the neighbors of router 54. Router 54's neighbors are switch 58 and router 52. By trying to communicate with the neighbors, host 51 determines that router 52 is active 206 and that switch 54 is not active. Therefore, router 54 is

identified as another root cause of the failure in 207 and an indication to that effect is output in 208.

Thus, as illustrated in Fig. 4, a single outage may have more than one root cause -- in that case, both routers 54 and 56. The invention will also identify failed devices in outages that have more than two root causes.

As described above, the illustrated embodiment relies on tables in a device to provide network addresses of the device's neighbors. If a device does not include such a table, as may be the case for some types of switches, it may be possible to deduce the presence of such devices, at least to a degree.

For example, if switches 10 to 12 do not respond with neighbor addresses in 302 (Fig. 3), but router 9 does, switch 12 and router 9 will appear disconnected from network 1 in the network neighbor table. However, since router 9 provided the requested information, this cannot be the case. Knowing this, and that router 9 is connected to switch 12 (based on router 9's response in 302), it can be deduced that there is at least one device between switch 12 and another device on network 1.

Note that the invention is not limited to the specific hardware and software configurations described above. For example, any tables that contain device neighbor addresses can be used in place of MIB II tables. TL1 tables are an example. Similarly, protocols other than IP, and network addresses other than IP and MAC addresses, can be used in the invention. The invention can also be used with networks having different router and switch configurations than those shown in Figs. 1 and 4. The invention can be implemented at different nodes of a network. For example, in the network of Fig. 1, the invention can be implemented

from host 2 (as described), from host 8, or from any other devices (not shown) at nodes of the network.

The present invention has been described with respect to a particular illustrative embodiment. It is to be understood that the invention is not limited to the above-described embodiment and modifications thereto, and that various changes and/or modifications are within the scope of the appended claims.

What is claimed is: